基于混合策略改进的鲸鱼优化算法*

何　庆[a, b†]，魏康园[a, b]，徐钦帅[a, b]

(贵州大学 a. 大数据与信息工程学院; b. 贵州省公共大数据重点实验室, 贵阳 550025)

摘　要：针对传统鲸鱼优化算法收敛速度慢、易陷入局部最优等问题，提出一种基于混合策略改进的鲸鱼优化算法。首先，引入非线性调整策略改进收敛因子，平衡算法的全局探索与局部开发能力并加快算法收敛速度；然后，将自适应权重系数引入鲸鱼位置更新式中，从而提高算法的寻优精度；最后，结合人工蜂群算法的 limit 阈值思想，使算法能够有效跳出局部最优，改善算法早熟收敛现象。通过对 14 个基准测试函数在不同维度上的仿真实验表明，改进算法具有较高的寻优精度和较快的收敛速度。

关键词：鲸鱼优化算法；非线性收敛因子；自适应权重系数；limit 阈值

中图分类号：TP301.6　　doi: 10.3969/j.issn.1001-3695.2018.07.0382

Mixed strategy based improved whale optimization algorithm

He Qing[a, b†], Wei Kangyuan[a, b], Xu Qinshuai[a, b]

(*a. College of Big Data & Information Engineering, b. Guizhou Provincial Key Laboratory of Public Big Data Guizhou University, Guiyang* 550025, *China*)

Abstract: In order to solve the disadvantage of the traditional whale optimization algorithm, which is slow convergence and easy to fall into local optimum, this paper proposed a mixed strategy based whale optimization algorithm. Firstly, it introduced the nonlinear adjustment strategy to modify the convergence factor, balance the exploration and exploitation capability and accelerate the convergence speed. Then, by introducing an adaptive weighted coefficient into the position update formula of whales to improve the search precision of the algorithm. Finally, it combined the limit threshold idea of artificial bee colony algorithm to effectively jump out of the local optimum and prevent premature convergence. The results show that the proposed algorithm has better search precision and convergence speed through experiments on different dimensions of 14 benchmark functions.

Key words: whale optimization algorithm; nonlinear convergence factor; adaptive weighted coefficient; limit threshold

0　引言

鲸鱼优化算法[1]（whale optimization algorithm，WOA）是 Seyedali Mirjalili 教授等于 2016 年提出的一种模拟鲸鱼群体捕食行为的启发式优化算法。该算法具有原理简单、易于实现、参数较少等特点，且研究发现，在对基准测试函数进行优化时，传统 WOA 算法的优化精度与收敛速度均明显优于粒子群优化算法（particle swarm optimization，PSO）和引力搜索算法（gravitational search algorithms，GSA）[1]。但是，与其他元启发式优化算法相类似，传统 WOA 算法同样存在收敛速度慢、早熟收敛、易陷入局部最优等问题，基于此，近年来国内外学者实现了许多有效的改进 WOA 算法，如 Kaur 等人[2]利用混沌映射优化 WOA 算法中的更新概率 p，提出一种 CWOA 算法，并通过基准测试函数的测试，验证了算法具有较高的收敛速度；Mafarja 等人[3]通过将模拟退火算法和鲸鱼优化算法进行融合，来优化算法的寻优精度，提高全局搜索能力，并在公开测试 UCI 库中的 18 个数据集的实验中取得了良好的结果；Mohamed 等人[4]利用反向学习进行初始化，增强算法对搜索空间的探索，提出了一种 OBWOA 算法，并将 OBWOA 算法应用到三种不同的二极管模型中，来估算太阳能电池的参数，通过实验验证该方法具有良好的探测能力；张永等人[5]首先通过分段 Logistic 混沌映射产生混沌序列对种群进行初始化，来维持初始种群多样性，同时引入分段自适应权重平衡算法全局探索和局部开发能力，提出了 MWOA 算法，并在 6 个基准测试函数的测试，

收稿日期：2018-07-19；**修回日期：**2018-09-01　**基金项目：**贵州省科技计划项目重大专项资助项目（黔科合重大专项字［2018］3002）；贵州省公共大数据重点实验室开放课题（2017BDKFJJ004）；贵州省教育厅青年科技人才成长项目（黔科合 KY 字［2016］124）；贵州大学培育项目（黔科合平台人才［2017］5788）

作者简介：何庆（1982-），男（通信作者），贵州贵阳人，副教授，博士，主要研究方向为大数据应用、人工智能（qhe@gzu.edu.cn）；魏康园（1991-），女，陕西渭南人，硕士研究生，主要研究方向为数据挖掘、进化计算；徐钦帅（1994-），男，山东滕州人，硕士研究生，主要研究方向为机器学习、进化计算.

验证了算法的性能；龙文等[6]为求解大规模复杂优化问题，首先利用对立学习策略进行种群位置初始化，设计一种非线性收敛因子，协调算法的探索和开发能力，同时引入多样性变异操作，改善算法早熟收敛现象。然而，尽管已有研究对传统 WOA 算法的寻优效果有所提高，但对于平衡算法全局探索和局部开发能力、算法收敛速度慢、易陷入局部最优等问题仍需深入研究。

综上所述，本文针对鲸鱼优化算法收敛速度慢、易陷入局部最优、发生早熟收敛等问题，首先对鲸鱼优化算法中的收敛因子进行改进，加快算法收敛速度、平衡算法探索和开发能力；然后引入自适应权重系数提高算法的寻优精度；最后针对鲸鱼优化算法易陷入局部最优的问题，结合人工蜂群算法（artificial bee colony, ABC）[7]-[9]较强的全局探索能力，引入阈值思想，避免算法陷入局部最优解。

1　鲸鱼优化算法

WOA 是一种模拟鲸鱼捕食行为的元启发式优化算法，其捕食行为成为泡泡网捕食方法[10]，分为三个阶段：包围猎物、泡网攻击、搜寻猎物。

在 WOA 算法中，假设鲸鱼种群规模表示为 N，搜索空间的维度为 Dim，则第 i 只鲸鱼在第 d 维的位置可以表示为：$X_i=\left(X_i^1,X_i^2,\cdots,X_i^{Dim}\right),i=1,2,\cdots,N$，猎物的位置代表问题的全局最优解。

1)包围猎物

在 WOA 中，鲸鱼识别猎物的位置，并将它们包围起来，然而鲸鱼无法提前获知猎物的位置。因此，假设当前的最优位置是目标猎物，群体中的其他个体均向最优位置移动，利用如下公式更新位置：

$$\vec{D}=\left|\vec{C}\cdot\vec{X}^*(t)-\vec{X}(t)\right| \tag{1}$$

$$\vec{X}(t+1)=\vec{X}^*(t)-\vec{A}\cdot\vec{D} \tag{2}$$

其中：t 表示为当前迭代次数；$\vec{X}^*(t)=\left(X_1^*,X_2^*,\cdots,X_{Dim}^*\right)$ 表示猎物位置；$\vec{X}(t)$ 表示当前鲸鱼位置；$\vec{A}\cdot\vec{D}$ 表示包围步长，其中 $\vec{A}$ 和 $\vec{C}$ 定义如下：

$$\vec{A}=2\vec{a}\cdot rand-\vec{a} \tag{3}$$

$$\vec{C}=2\cdot rand \tag{4}$$

其中：$rand$ 表示 $[0,1]$ 之间的随机数；$\vec{a}$ 表示随着迭代次数的增加从 2 线性递减到 0 的收敛因子，表示为：

$$a=\left(2-{2t}/{T_{\max}}\right) \tag{5}$$

其中：t 表示为当前迭代次数；$T_{\max}$ 表示为最大迭代次数。

2)泡网攻击

在 WOA 算法中，设计两种方法来描述鲸鱼的捕食行为，分别为：收缩包围机制、螺旋更新位置。

a）收缩包围机制：通过减小公式（3）和（5）中的收敛因子 a 来实现。

b）螺旋更新位置：首先计算鲸鱼个体与当前最优位置之间的距离，然后模拟鲸鱼以螺旋的方式捕获食物，其数学模型可以表示为：

$$\vec{X}(t+1)=\vec{D}'\cdot e^{bl}\cdot\cos(2\pi l)+\vec{X}^*(t) \tag{6}$$

其中：$\vec{D}'=\left|\vec{X}^*(t)-\vec{X}(t)\right|$ 表示第 i 只鲸鱼和当前最优位置之间的距离；b 是用来限定对数螺旋形式的常量系数，文本取 1；l 表示 $[-1,1]$ 之间的随机数。

值得注意的是，鲸鱼以螺旋形式包围猎物的同时，还需收缩包围圈，因此，为了实现这种同步模型，则选择相同概率 p 进行收缩包围机制和螺旋位置更新，其数学模型表示如下：

$$\vec{X}(t+1)=\begin{cases}\vec{X}^*(t)-\vec{A}\cdot\vec{D} & if\quad p<0.5\\ \vec{D}'\cdot e^{bl}\cdot\cos(2\pi l)+\vec{X}^*(t) & if\quad p\geq 0.5\end{cases} \tag{7}$$

3)搜寻猎物

当 $|A|\geq 1$ 时，随机选择鲸鱼迫使其远离参考鲸鱼，来找到一个更优的猎物，以便增强算法的全局探索能力，其数学模型表示如下：

$$\vec{D}=\left|\vec{C}\cdot\vec{X}_{rand}-\vec{X}(t)\right| \tag{8}$$

$$\vec{X}(t+1)=\vec{X}_{rand}-\vec{A}\cdot\vec{D} \tag{9}$$

其中：$\vec{X}_{rand}$ 表示随机选取鲸鱼的位置向量。

2　混合策略改进的鲸鱼优化算法

本文针对鲸鱼优化算法收敛速度慢、易陷入局部最优、发生早熟收敛等问题，引入三种改进策略，提出一种基于混合策略改进的鲸鱼优化算法（mixed strategy based improved whale optimization algorithm, MS-WOA）。

基于混合策略改进的鲸鱼优化算法流程图如图 1 所示：

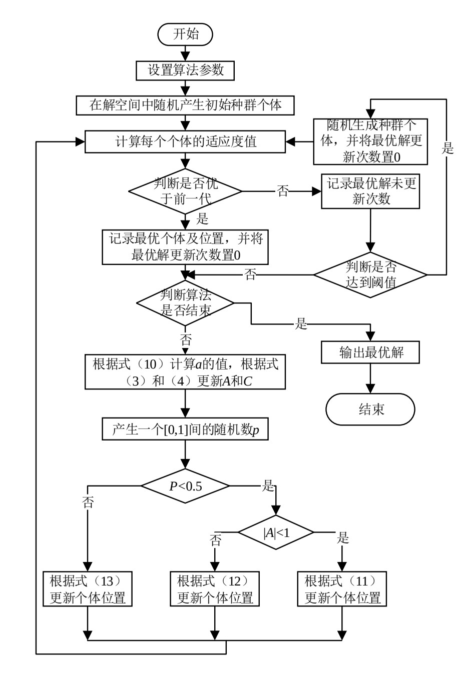


图 1　MS-WOA 算法流程图

2.1　非线性收敛因子

通过分析传统 WOA 算法的基本原理可知，$\vec{A}\cdot\vec{D}$ 为包围步

长，且算法的全局探索和局部开发能力主要依赖于参数 $\vec{A}$，根据式（3）（5）可知，参数 $\vec{A}$ 的取值主要取决于收敛因子 $\vec{a}$ 的变化。因此，收敛因子 $\vec{a}$ 对于寻找算法最优解至关重要，较大的收敛因子能够提供较强的全局探索能力，避免陷入局部最优值；而较小的收敛因子使得算法具有较强的局部开发能力，能够加快算法的收敛速度[1]。然而，在传统 WOA 算法中，收敛因子的变化是随着迭代次数的增加而线性递减，这种变化易使得算法收敛速度过慢。因此，受文献[11，12]中改进策略的启发，本文在不改变原始收敛因子变化趋势的情况下，引入非线性调整策略，以便保证算法的全局探索和局部开发能力的同时，加快算法的收敛速度。具体公式如下：

$$a=\left(2-{2t}/{T_{\max}}\right)\left(1-{t^3}/{T_{\max}^3}\right) \tag{10}$$

其中：t 表示为当前迭代次数；$T_{\max}$ 表示为最大迭代次数。非线性递减的收敛因子 a 在算法早期迭代时生成较大参数 $\vec{A}$，能够更加有效的提升全局探索能力，同时加快算法收敛速度；在后期迭代时生成较小参数 $\vec{A}$，能够有效提升局部开发能力。

2.2　自适应权重系数

在 WOA 算法中，猎物的位置代表优化问题的最优解，然而，在传统 WOS 算法位置更新公式（2）、（6）、（9）中猎物的位置 $\vec{X}^*(t)$ 都未被充分利用。因此，本文在位置更新中引入自适应权重，以便最优解能被更充分利用，从而提高算法的寻优精度。定义如下：

$$\vec{X}(t+1)=\left({t^3}/{T_{\max}^3}\right)\cdot\vec{X}^*(t)-\vec{A}\cdot\vec{D},\ |A|<1\quad p<0.5 \tag{11}$$

$$\vec{X}(t+1)=\left({t^3}/{T_{\max}^3}\right)\cdot\vec{X}_{rand}-\vec{A}\cdot\vec{D},\ |A|\geq 1\ \ p<0.5 \tag{12}$$

$$\vec{X}(t+1)=\vec{D}'\cdot e^{bl}\cdot\cos(2\pi l)+\left(1-{t^3}/{T_{\max}^3}\right)\cdot\vec{X}^*(t),\ p\geq 0.5 \tag{13}$$

其中：${t^3}/{T_{\max}^3}$ 表示猎物位置的自适应权重，随着迭代次数的增加权重系数不断增加，意味着，在 WOA 算法中包围猎物阶段和随机搜索猎物阶段，即当 $p<0.5$ 时，在猎物位置中引入自适应权重 ${t^3}/{T_{\max}^3}$，以便寻优问题中的最优解被充分利用；同时，在螺旋更新位置阶段采用的自适应权重为 $1-{t^3}/{T_{\max}^3}$，随着迭代次数的增加，鲸鱼将不断接近猎物，且此时采用较小的权重改变猎物位置，以便提高算法局部开发能力，从而提高算法的寻优精度。

2.3　limit 阈值

通过分析传统 WOA 算法可知，随着算法的迭代进化，鲸鱼将逐渐向适应度较优的个体聚集，从而使得种群分布收缩，多样性减小，导致算法易陷入局部最优。针对群智能算法易陷入局部最优的共性问题，文献[13，14]对于人工蜂群算法，通过观察最优解未更新的次数是否大于 limit 阈值，从而使得算法有效跳出局部最优解；文献[15]将 limit 阈值思想引入蝙蝠算法中，来避免陷入局部极值；文献[16]通过将局部搜索能力强的果蝇优化算法与全局搜索能力强的人工蜂群算法进行融合，引入阈值思想，来改进节点定位误差函数。因此，本文基于人工蜂群算法的原理，通过引入 limit 阈值思想，限定算法陷入局部最优解的次数，以改善算法易陷入局部极值的问题。然而，阈值 limit 的取值将会影响算法寻优结果，若取值过大则对算法改进甚微；反之，取值过小则算法局部开发能力被减弱，且算法更新频繁。通过多次测试，本文将阈值 limit 设定为 60 较为理想。

2.4　改进算法复杂度分析

已知算法的种群规模为 N，最大迭代次数为 $T_{\max}$，搜索空间维度为 Dim。根据改进算法流程图可知，引入的非线性调整策略和自适应权重系数，增加了 $O(T_{\max}\cdot N\cdot Dim)$ 的运算量；同时，为避免算法陷入局部最优，引入的阈值思想，增加一个内层循环，则算法的时间复杂度可表示为

$$T(n)=O(f(n))=O\left(T_{\max}\cdot N^2\cdot(N-1)\cdot Dim\right) \tag{14}$$

另外，空间复杂度 $S(n)$ 主要受种群规模和搜索空间维度的影响，可表示为

$$S(n)=O(f(n))=O(N\cdot Dim) \tag{15}$$

其中：本文将采用 4 组不同维度进行仿真实验，且维度越大，占用空间存储空间越大，空间复杂度越高。

3　实验结果与分析

本文仿真实验基于 Windows 7（64bit）操作系统，Intel(R) Core(TM) i5-6500 CPU、3.20GHz 主频及 8GB 内存，编程采用 MATLAB R2015b 软件。为了验证本文提出的 MS-WOA 算法的性能，引入 14 个基准测试函数[17]（如表 1 所示），其中 F_1~F_6 是连续单模态函数、F_7~F_{12} 是复杂非线性多模态函数、F_{13}~F_{14} 是固定低维函数。

本文从两个方面对算法进行测试：a)针对三个不同维度选取不同迭代次数对算法进行测试，并与灰狼优化算法（grey wolf optimizer，GWO）[18]、正弦余弦算法（sin cosin algorithm，SCA）[19]、文献[4,6]进行比较，分析算法的寻优精度和收敛速度；b)针对固定低维函数进行测试，并与参考文献[6]进行寻优精度和性能对比。

3.1　30 维函数优化性能测试

在 30 维搜索空间中测试 MS-WOA 算法的优化性能，设定种群规模为 30，最大迭代次数为 500，基准测试函数为 F_1-F_{12}。表 2 展示了 GWO 算法、SCA 算法、WOA 算法、IWOA 算法[6]和 MS-WOA 算法进行 30 次独立实验后的测试结果，分别从平均解和标准差两个方面进行对比。图 2 为 GWO 算法、SCA 算法、WOA 算法和 MS-WOA 算法优化基准测试函数的收敛过程对比图。

从表 2 可以看出，针对 12 个基准测试函数，MS-WOA 算法对大部分函数的寻优精度比 IWOA 算法高出多个数量级，而 MS-WOA 算法的寻优精度更是远远优于传统 GWO 算法、SCA 算法、WOA 算法。其中在连续单模态的函数求解中，对于函数 F_1 取得了理论最优解，对于函数 F_2、F_3、F_4 标准差均取到了 0，接近于其理论值，对于函数 F_6 的优化也略优于 IWOA 算法；

只是对于函数 F_5 的优化效果略劣于 IWOA 算法，但通过图 2(e)可知，收敛速度相对于传统 GWO、SCA、WOA 算法具有显著提高。同时由表 2 和图 2(f)-(l)可知，针对具有多个局部极值的多模态函数 F_7~F_{12} 的优化求解精度与速度均显著优于其他算法。

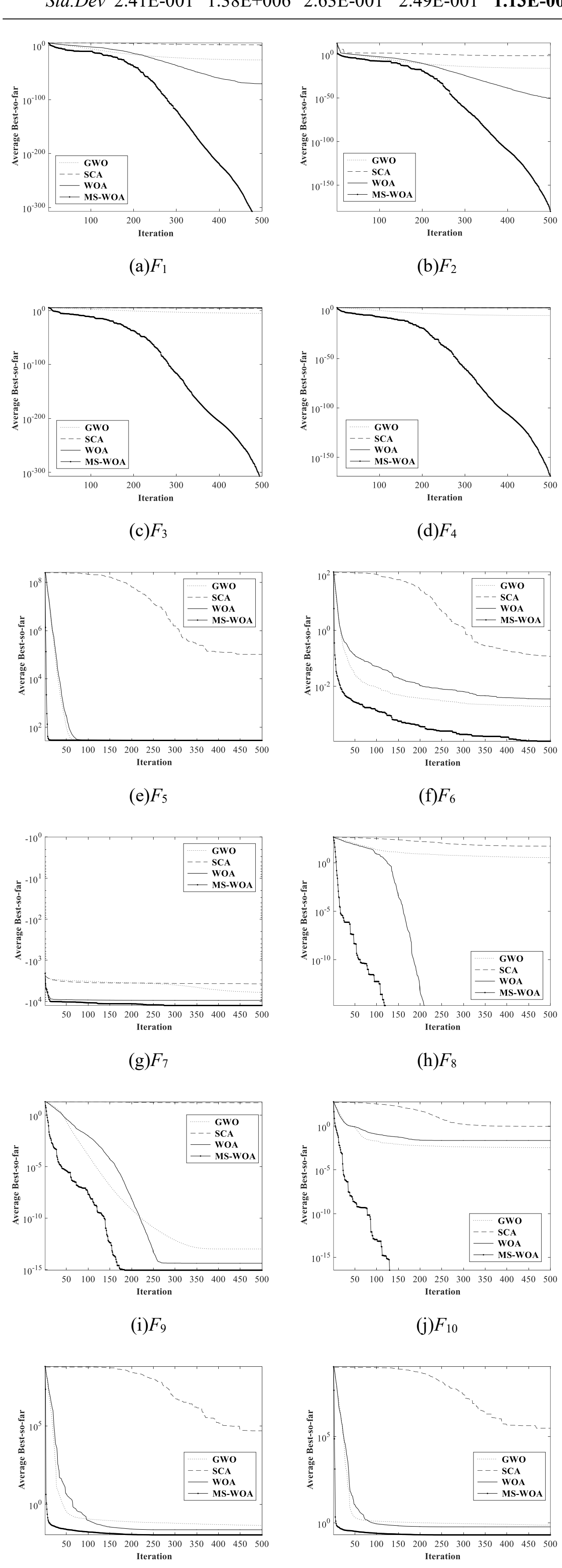
表 1　基准测试函数

函数名	表达式	维度	范围	理论最优
Sphere	$F_1=\sum_{i=1}^{Dim}x_i^2$	30/10/200	[-100,100]	0
Schwefel 2.22	$F_2=\sum_{i=1}^{Dim}\lvert x_i\rvert+\prod_{i=1}^{Dim}\lvert x_i\rvert$	30/10/200	[-10,10]	0
Schwefel 1.2	$F_3=\sum_{i=1}^{Dim}\left(\sum_{j=1}^{i}x_i\right)^2$	30/10/200	[-100,100]	0
Schwefel 2.21	$F_4=\max_i\{\lvert x_i\rvert,1\le i\le D\}$	30/10/200	[-100,100]	0
Rosenbrock	$F_5=\sum_{i=1}^{Dim-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	30	[-30,30]	0
Quartic	$F_6=\sum_{i=1}^{Dim}i\cdot x_i^4+random[0,1)$	30/10/200	[-1.28,1.28]	0
Schwefel 2.26	$F_7=\sum_{i=1}^{Dim}-x_i\sin\left(\sqrt{x_i}\right)$	30	[-500,500]	-1.2569.5
Rastrigin	$F_8=\sum_{i=1}^{Dim}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30/10/200	[-5.12,5.12]	0
Ackley	$F_9=-20\exp\left(-0.2\sqrt{\frac{1}{Dim}\sum_{i=1}^{Dim}x_i^2}\right)-\exp\left(\frac{1}{Dim}\sum_{i=1}^{Dim}\cos\left(2\pi x_i\right)\right)+20+e$	30/10/200	[-32,32]	0
Griewank	$F_{10}=\frac{1}{4000}\sum_{i=1}^{Dim}x_i^2-\prod_{i=1}^{Dim}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30/10/200	[-600,600]	0
Penalized	$F_{11}=\frac{\pi}{Dim}\left\{10\sin\left(\pi y_1\right)+\sum_{i=1}^{Dim-1}\left(y_i-1\right)^2\left[1+10\sin^2\left(\pi y_{i+1}\right)\right]+\left(y_{Dim}-1\right)^2\right\}+\sum_{i=1}^{Dim}u(x_i,10,100,4)$	30	[-50,50]	0
Generalized Penalized	$F_{12}=0.1\left\{\sin^2\left(3\pi x_1\right)+\sum_{i=1}^{Dim}\left(x_i-1\right)^2\left[1+\sin^2\left(3\pi x_i+1\right)\right]+\left(x_{Dim}-1\right)^2\left[1+\sin^2\left(3\pi x_{Dim}\right)\right]\right\}+\sum_{i=1}^{Dim}u\left(x_i,5,100,4\right)$	30	[-50,50]	0
Shekel	$F_{13}=\left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}\left(x_i-a_{ij}\right)^6}\right]^{-1}$	2	[-65.536,65.536]	1
Kowalik	$F_{14}=\sum_{i=1}^{11}\left[a_i-\frac{x_1\left(b_i^2+b_ix_2\right)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[-5,5]	0.0003

表 2　MS-WOA 与其他算法优化基准函数对比（*Dim*=30）

函数		GWO	SCA	WOA	IWOA[6]	MS-WOA
F_1	*Mean*	2.28E-027	1.99E+001	1.63E-071	6.54E-125	**0.00E+000**
	Std.Dev	5.24E-027	4.43E+001	8.47e-071	6.80E-125	**0.00E+000**
F_2	*Mean*	1.18E-016	2.73E-002	5.02E-051	2.15E-073	**1.31E-180**
	Std.Dev	1.12E-016	5.52E-002	2.15E-050	3.64E-073	**0.00E+000**
F_3	*Mean*	5.67E-006	8.21E+003	4.55E+004	1.56E-023	**5.63E-322**
	Std.Dev	1.24E-005	5.35E+003	1.26E+004	1.81E-023	**0.00E+000**
F_4	*Mean*	7.16E-007	3.24E+001	4.90E+001	7.06E-007	**9.64E-170**
	Std.Dev	7.85E-007	1.41E+001	2.76E+001	2.18E-006	**0.00E+000**
F_5	*Mean*	2.74E+001	1.03E+005	2.80E+001	**2.73E+001**	2.81E+001
	Std.Dev	8.47E-001	2.48E+005	4.22E-001	**2.15E-001**	3.64E-001
F_6	*Mean*	1.85E-003	1.16E-001	3.47E-003	2.42E-004	**1.05E-004**
	Std.Dev	8.57E-004	8.54E-002	3.80E-003	4.41E-004	**7.57E-005**
F_7	*Mean*	-6.00E+003	-3.73E+003	-9.44E+003	-1.14E+004	**-1.26E+004**
	Std.Dev	1.03E+003	2.74E+002	8.25E+002	1.13E+002	**1.55E+001**
F_8	*Mean*	3.36E+000	5.11E+001	0.00E+000	0.00E+000	**0.00E+000**
	Std.Dev	4.26E+000	3.95E+001	0.00E+000	0.00E+000	**0.00E+000**
F_9	*Mean*	1.04E-013	1.61E+001	4.20E-015	3.02E-015	**8.88E-016**
	Std.Dev	1.94E-014	7.71E+000	2.62E-015	1.95E-015	**0.00E+000**
F_{10}	*Mean*	3.53E-003	9.49E-001	2.31E-002	0.00E+000	**0.00E+000**
	Std.Dev	7.75E-003	4.63E-001	7.31E-002	0.00E+000	**0.00E+000**

F_{11}	*Mean*	4.65E-002	4.76E+004	2.43E-002	8.76E-002	**1.15E-002**
	Std.Dev	2.98E-002	1.48E+005	1.47E-002	1.20E-002	**5.43E-003**
F_{12}	*Mean*	7.38E-001	3.02E+005	5.84E-001	4.66E-001	**2.00E-001**
	Std.Dev	2.41E-001	1.38E+006	2.63E-001	2.49E-001	**1.13E-001**



(a)F_1　(b)F_2　(c)F_3　(d)F_4　(e)F_5　(f)F_6　(g)F_7　(h)F_8　(i)F_9　(j)F_{10}　(k)F_{11}　(l)F_{12}

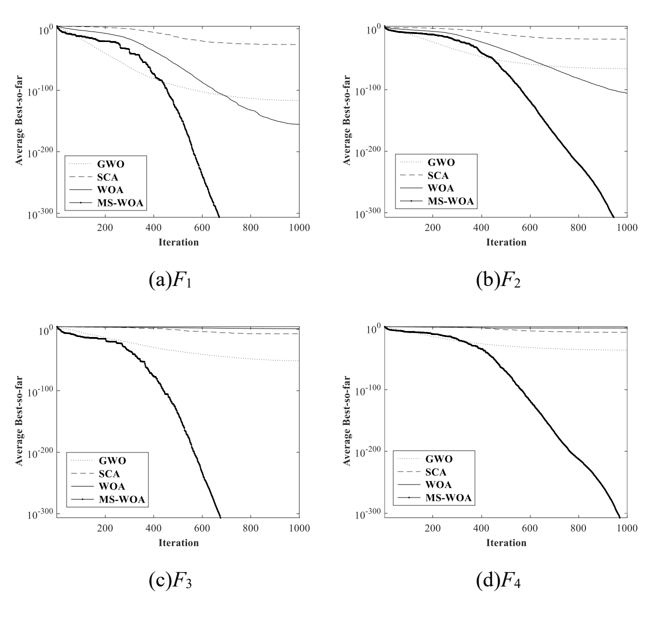
图 2　MS-WOA 与其他算法优化基准测试函数收敛曲线（*Dim*=30）

3.2　10 维函数优化性能测试

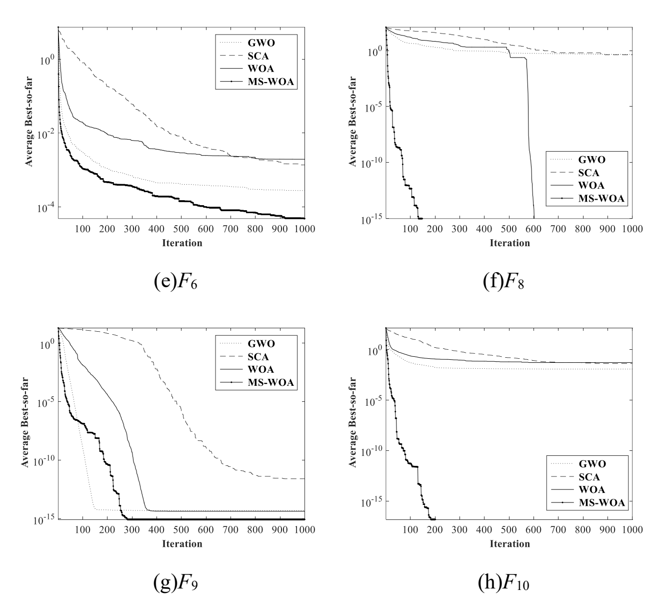
在 10 维搜索空间中测试 MS-WOA 算法的优化性能，设定种群规模为 30，最大迭代次数为 1000，基准测试函数为函数 F_1-F_{12} 中 8 个经典函数，分别为：F_1、F_2、F_3、F_4、F_6、F_8、F_9、F_{10}。分别使用 GWO 算法、SCA 算法、WOA 算法和 MS-WOA 算法对选取的 8 个函数进行 30 次独立实验，并与文献[4]中 OBWOA 算法进行对比，如表 3 所示。图 3 对应上述算法优化基准测试函数时收敛过程对比图。

表 3　MS-WOA 与其他算法优化基准函数对比（*Dim*=10）

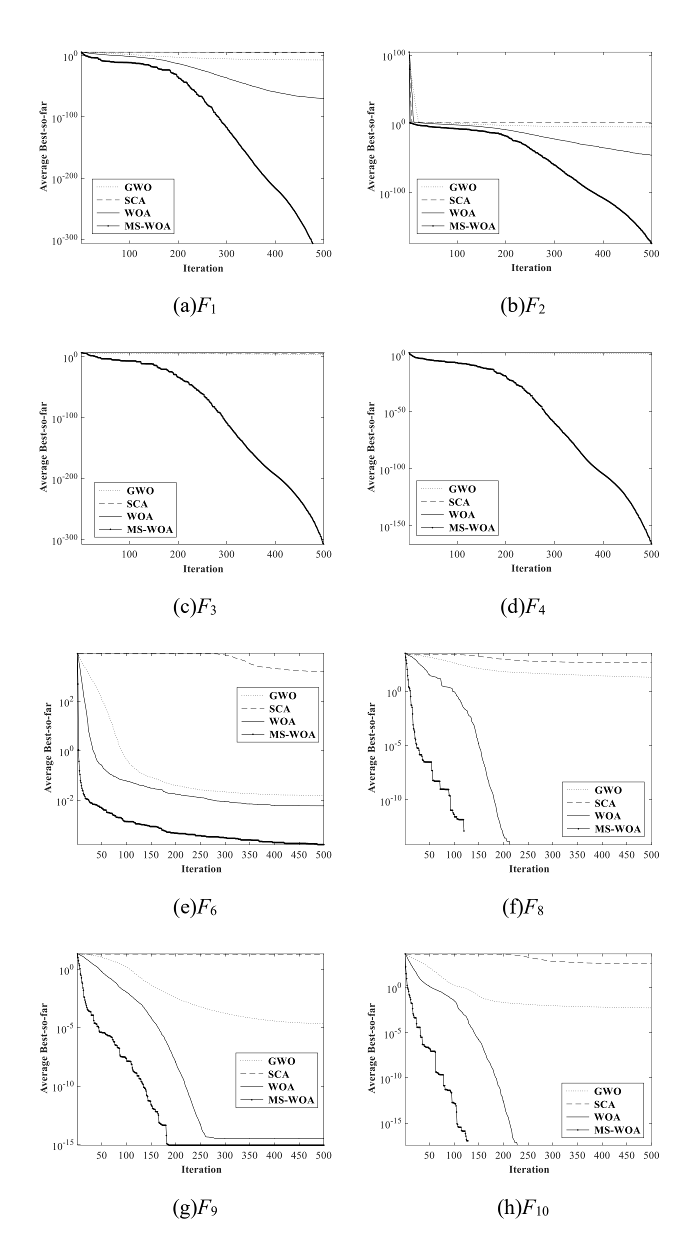
函数		GWO	SCA	WOA	OBWOA[4]	MS-WOA
F_1	*Mean*	1.87E-117	1.50E-026	2.91E-156	0.00E+000	**0.00E+000**
	Std.Dev	5.49E-117	8.09E-026	1.51E-155	0.00E+000	**0.00E+000**
F_2	*Mean*	1.31E-066	1.31E-018	2.50E-106	4.90E-301	**0.00E+000**
	Std.Dev	3.67E-066	5.74E-018	1.35E-105	0.00E+000	**0.00E+000**
F_3	*Mean*	6.98E-052	3.10E-008	6.63E+000	1.48E+005	**0.00E+000**
	Std.Dev	3.02E-051	1.66E-007	1.10E+001	1.09E+005	**0.00E+000**
F_4	*Mean*	7.17E-037	5.90E-008	3.02E-001	6.42E-051	**0.00E+000**
	Std.Dev	3.35E-036	1.91E-007	1.18E+000	2.99E-050	**0.00E+000**
F_6	*Mean*	2.75E-004	1.36E-003	1.94E-003	1.75E-002	**4.77E-005**
	Std.Dev	1.87E-004	9.03E-004	2.42E-003	1.37E-002	**5.12E-005**
F_8	*Mean*	4.61E-001	4.19E-001	0.00E+000	0.00E+000	**0.00E+000**
	Std.Dev	1.47E+000	2.29E+000	0.00E+000	0.00E+000	**0.00E+000**
F_9	*Mean*	4.91E-015	2.63E-012	4.44E-015	8.88E-016	**8.88E-016**
	Std.Dev	1.23E-015	1.44E-011	2.47E-015	0.00E+000	**0.00E+000**
F_{10}	*Mean*	1.18E-002	7.19E-002	5.24E-002	0.00E+000	**0.00E+000**
	Std.Dev	1.54E-002	1.55E-001	1.10E-001	0.00E+000	**0.00E+000**



(a)F_1　(b)F_2　(c)F_3　(d)F_4



(e)F_6　(f)F_8　(g)F_9　(h)F_{10}



(a)F_1　(b)F_2　(c)F_3　(d)F_4

图 3　MS-WOA 与 WOA 优化基准测试函数收敛曲线（*Dim*=10）

观察表 3 可知，当最大迭代次数取 1000 时，MS-WOA 算法对于选取的函数均明显优于其他算法，且除函数 F_6 均取得其理论最优值，然而 MS-WOA 算法对于函数 F_6 的寻优精度比 OBWOA 算法高出 3 个数量级。且由图 3 可知，MS-WOA 算法的收敛速度相比于传统 SCA、WOA 算法均有明显提高；然而相比于传统 GWO 算法，在函数 F_1、F_2、F_{10} 的优化中，MS-WOA 算法前期的收敛速度略劣于 GWO 算法。

3.3　200 维函数优化性能测试

在 200 维搜索空间中测试 MS-WOA 算法的优化性能，设定设定种群规模为 30，最大迭代次数为 500，选取同 3.2 节相同的基准测试函数，独立运行 30 次，并与文献[6]中 IWOA 算法进行对比，测试结果如表 4 所示（“—”表示参考文献未给出），收敛过程对比图如图 4 所示。

表 4　MS-WOA 与其他算法优化基准函数对比（*Dim*=200）

函数		GWO	SCA	WOA	IWOA[6]	MS-WOA
F_1	*Mean*	9.74E-008	5.42E+004	8.52E-071	2.62E-116	**0.00E+000**
	Std.Dev	5.80E-008	2.38E+004	2.48E-070	1.50E-116	**0.00E+000**
F_2	*Mean*	2.90E-005	3.07E+001	2.50E-046	1.68E-067	**6.59E-175**
	Std.Dev	6.58E-006	1.84E+001	1.37E-045	1.95E-067	**0.00E+000**
F_3	*Mean*	2.21E+004	1.08E+006	5.28E+006	—	**4.74E-312**
	Std.Dev	1.20E+004	1.77E+005	1.27E+006	—	**0.00E+000**
F_4	*Mean*	2.63E+001	9.63E+001	7.93E+001	—	**1.12E-166**
	Std.Dev	8.02E+000	1.53E+000	2.26E+001	—	**0.00E+000**
F_6	*Mean*	1.54E-002	1.60E+003	5.82E-003	1.44E-003	**1.58E-004**
	Std.Dev	4.78E-003	5.18E+002	7.74E-003	1.77E-003	**1.44E-004**
F_8	*Mean*	2.14E+001	4.76E+002	0.00E+000	0.00E+000	**0.00E+000**
	Std.Dev	1.23E+001	1.75E+002	0.00E+000	0.00E+000	**0.00E+000**
F_9	*Mean*	2.34E-005	1.85E+001	3.26E-015	8.88E-016	**8.88E-016**
	Std.Dev	5.17W-006	4.28E+000	2.35E-015	0.00E+000	**0.00E+000**
F_{10}	*Mean*	5.57E-003	4.21e+002	0.00E+000	0.00E+000	**0.00E+000**
	Std.Dev	1.46E-002	2.45e+002	0.00E+000	0.00E+000	**0.00E+000**

(e)F_6　(f)F_8　(g)F_9　(h)F_{10}

图 4　MS-WOA 与 WOA 优化基准测试函数收敛曲线（*Dim*=200）

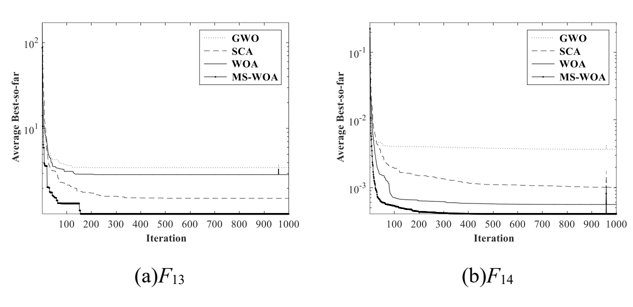
从实验结果可以看出，对于高维函数的优化求解，MS-WOA 算法的求解精度和收敛速度均明显优于其他算法。

3.4　固定低维函数优化性能测试

在固定维度搜索空间中测试 MS-WOA 算法的优化性能，设定种群规模为 30，最大迭代次数为 1000，基准测试函数为 F_{13}、F_{14}，独立运行 30 次，测试结果为表 5 所示，收敛过程对比图如图 5 所示。从实验结果可以看出，对于固定低维函数的测试，MS-WOA 算法的求解精度和收敛速度仅略优于其他算法。

表 5　MS-WOA 与其他算法优化基准函数对比（固定低维）

函数		GWO	SCA	WOA	OBWOA[6]	MS-WOA
F_{13}	*Mean*	3.49E+000	1.53E+000	2.89E+000	4.07E+000	**1.01E+000**
	Std.Dev	3.43E+000	8.92E-001	3.64E+000	3.53E+000	**1.26E-001**
F_{14}	*Mean*	3.70E-003	1.00E-003	5.60E-004	4.63E-004	**4.06E-004**
	Std.Dev	7.60E-003	3.53E-004	3.20E-004	4.00E-004	**1.05E-004**



(a)F_{13} (b)F_{14}

图 5 MS-WOA 与 WOA 优化基准测试函数收敛曲线（固定低维）

上述实验结果表明，本文引入非线性收敛因子策略，能够明显加快算法收敛速度，平衡算法全局探索与局部开发能力；同时，提出自适应权重方法，能够有效提高算法寻优精度，并当维度为 10，最大迭代次数为 1000 时，已有很多函数取得理论最优值；最后，引入阈值思想，能够较好的避免算法陷入局部最优，出现早熟收敛现象。

综上所述，针对基准函数优化问题，本文提出的 MS-WOA 算法在寻优精度和稳定性方面均优于其他算法；而在收敛速度方面，除维度为 10 时略劣于 GWO 算法外，该算法均明显优于其他算法，验证了 MS-WOA 算法中改进策略的有效性和优越性。

4 结束语

本文首先将非线性调整策略引入传统 WOA 算法中对收敛因子进行改进，平衡算法全局探索和局部开发能力并加快算法收敛速度；然后，提出一种自适应权重改进 WOA 算法中鲸鱼的位置更新公式，提高算法的寻优精度；最后，结合人工蜂群算法，将 limit 阈值思想引入 WOA 算法中，使得算法能够有效跳出局部最优，增强算法的全局探索能力。通过对 14 个基准测试函数在不同维度上的仿真实验，验证了本文所提出的 MS-WOA 算法相比于其他算法具有较高的寻优精度和较快的收敛速度。

参考文献：

[1] Mirjalili S, Lewis A. The whale optimization algorithm [J]. Advances in Engineering Software, 2016, 95: 51-67.

[2] Kaur G, Arora S. Chaotic whale optimization algorithm [J]. Journal of Computational Design and Engineering, 2018, 5: 275–284

[3] Mafarja M M, Mirjalili S. Hybrid whale optimization algorithm with simulated annealing for feature selection [J]. Neurocomputing, 2017, 260: 302-312.

[4] Elaziz M A, Oliva D. Parameter estimation of solar cells diode models by an improved opposition-based whale optimization algorithm [J]. Energy Conversion & Management, 2018, 171: 1843-1859.

[5] 张永，陈锋．一种改进的鲸鱼优化算法 [J]. 计算机工程，2018, 44 (3): 208-213, 219. (Zhang Yong, Chen Feng. A modified whale optimization algorithm [J]. Computer Engineering, 2018, 44 (3): 208-213, 219.)

[6] 龙文，蔡绍洪，焦建军，等．求解大规模优化问题的改进鲸鱼优化算法 [J]. 系统工程理论与实践，2017, 37 (11): 2983-2994. (Long Wen, Cai Shaohong, Jiao Jianjun, *et al.* Improved whale optimization algorithm for large scale optimization problems [J]. Systems Engineering-Theory & Practice, 2017, 37 (11): 2983-2994.)

[7] Karaboga D. An idea based on honey bee swarm for numerical optimization [D]. Kayseri: Erciyes University, 2005.

[8] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm [J]. Applied Soft Computing, 2008, 8 (1): 687-697.

[9] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm [J]. Applied Mathematics & Computation, 2009, 214 (1): 108-132.

[10] Watkins W A, Schevill W E. Aerial observation of feeding behavior in four baleen whales: eubalaena glacialis, balaenoptera borealis, megaptera novaeangliae, and balaenoptera physalus [J]. Journal of Mammalogy, 1979, 60 (1): 155-163.

[11] 魏政磊，赵辉，李牧东，等．控制参数值非线性调整策略的灰狼优化算法 [J]. 空军工程大学学报：自然科学版，2016, 17 (3): 68-72. (Wei Zhenglei, Zhao Hui, Li Mudong, *et al.* A grey wolf optimization algotithm based on nonlinear adjustment strategy of control parameter [J]. Journal of Air Force Engineering University: Natural Science Edition, 2016, 17 (3): 68-72.)

[12] 周敏，李太勇．粒子群优化算法中的惯性权值非线性调整策略 [J]. 计算机工程，2011, 37 (5): 204-206. (Zhou Min, Li Taiyong. Nonlinear adjustment strategy of inertia weight in particle swarm optimization algorithm [J]. Computer Engineering, 2011, 37 (5): 204-206.)

[13] 易正俊，韩晓晶．增强寻优能力的改进人工蜂群算法 [J]. 数据采集与处理，2013, 28 (6): 761-769. (Yi Zhengjun, Han Xiaojing. Modified artificial bee colony algorithm for search ability enhancement [J]. Journal of Data Acquisition and Processing, 2013, 28 (6): 761-769.)

[14] 邓高峰，叶金才，王国富，等．基于蜂群算法和新阈值函数的信号去噪算法 [J/OL]. 计算机应用研究, 2019, 36 (10) . (2018-07-09) [2018-07-15]. http://www. arocmag. com/article/02-2019-10-054. html. (Deng Gaofeng, Ye Jincai, Wang Guofu, *et al.* Signal denoising method based on bee colony algorithm and new threshold function [J/OL]. Application Research of Computers, 2019, 36 (10) . (2018-07-09) [2018-07-15]. http://www. arocmag. com/article/02-2019-10-054. html.)

[15] 杨菊婧，张达敏．基于改进 BA 算法的 K-means 聚类 [J]. 计算机应用研究，2018, 35 (5): 1454-1457. (Yang Juqing, Zhang Damin. K-means clustering algorithm based on improved BA algorithm [J]. Application Research of Computers, 2018, 35 (5): 1454-1457.)

[16] 徐同伟，何庆，吴意乐，等．基于自适应 ABC//FOA 融合定位算法研究 [J]. 传感技术学报, 2017, 30 (2): 278-283. (Xu Tongwei, He Qing, Wu Yile, *et al.* Research on the localization algorithm based on adaptive ABC//FOA fusion [J]. Chinese Journal of Sensors and Actuators, 2017, 30 (02): 278-283.)

[17] Hedar J. Test functions for unconstrained global optimization [EB/OL]. http://www-optima. amp. i. kyoto-u. ac. jp/member/student/hedar_files/Test

GO_files/Page364. html.

[18] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer [J]. Advances in Engineering Software, 2014, 69 (3): 46-61.

[19] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems [J]. Knowledge-Based Systems, 2016, 96: 120-133.